Exploiting language resources for semantic web annotations

Kaljurand, K ; Rinaldi, Fabio ; Dowdall, J ; Hess, M

Abstract: A large portion of the useful information on the web is in the form of unstructured natural language documents. Currently such documents are understandable to humans but not to software agents. One of the goals of the Semantic Web activity is to enrich a considerable number of web documents with annotations, which will then allow new generation search engines and novel web services to access those documents in a more intelligent fashion than currently possible. Currently the most reliable method of providing such semantic markup is via manual annotation, possibly based on predefined ontologies and with the support of specialized editors. In this paper we propose an approach for the automatic processing of textual documents to be published on the web, which can be used to automatically generate (some of) the semantic annotations. In particular, we focus on detecting the entities mentioned in the documents, their roles and relationships to other entities.

Exploiting Language Resources for Semantic Web Annotations

Kaarel Kaljurand, Fabio Rinaldi, James Dowdall, Michael Hess

Institute of Computational Linguistics, University of Zürich,
Winterthurerstrasse 190, CH-8057 Zürich, Switzerland
{kalju,rinaldi,dowdall,hess}@cl.unizh.ch

Abstract

A large portion of the useful information on the web is in the form of unstructured natural language documents. Currently such documents are understandable to humans but not to software agents. One of the goals of the Semantic Web activity is to enrich a considerable number of web documents with annotations, which will then allow new generation search engines and novel web services to access those documents in a more intelligent fashion than currently possible. Currently the most reliable method of providing such semantic markup is via manual annotation, possibly based on predefined ontologies and with the support of specialized editors. In this paper we propose an approach for the automatic processing of textual documents to be published on the web, which can be used to automatically generate (some of) the semantic annotations. In particular, we focus on detecting the entities mentioned in the documents, their roles and relationships to other entities.

1. Introduction

The documents available in the World Wide Web are mostly written in some natural language. As such, they are understandable only to humans. One of the directions of Semantic Web[1] research is about adding a layer to the documents that somehow formalizes their content, making it understandable also to software agents. The existence and functioning of such a layer enables automating several important knowledge management tasks (Rinaldi et al., 2003a). Such Semantic Web annotations can be seen as a way to mark explicitly the meaning of certain parts of the documents. They can be formalized and linked to ontologies using emerging web standards, such as OWL (McGuinness and van Harmelen (eds.), 2004). Various authors have addressed similar issues (Buitelaar and Declerck, 2003), which are particularly relevant in scaling up the Semantic Web (Handschuh and Staab, 2003).

At present day the semantic markup of documents is largely added manually. Although the human annotator can use specialized editors (Cimiano and Handschuh, 2003), which access predefined Ontologies and help to make certain annotation decisions, such work is time-consuming and can result in inconsistent annotation. Besides, it might require specific knowledge of Semantic Web standards, and certainly extra effort and special training of the annotators.

On the other hand, the semantic markup cannot yet be exploited to its full potential, which is another disincentive for annotators. Currently the only machines that crawl the web are search engines which see the documents as unordered sets of keywords. A critical mass of intelligent agents are needed to create interest in the web content providers. In the context of exponentially growing web, manual annotation is obviously an ineffective way to cope with the problem. In order to automate the process, a possible solution is to use some form of natural language processing.

In this paper we explore an NLP-based approach to automatically generate some of the semantic annotations to the web documents. Given the limited space available, we are forced to limit the description of our approach to a short summary (section 2.). A more detailed description of our work can be found in (Rinaldi et al., 2003b). In this paper we prefer to focus on some aspects of the problem discussed, namely the selection of a dependency based parser that serves our purpose (section 3.), and the combination of syntactic and semantic knowledge via a set of axioms (section 4.).

2. Description of the approach

In our approach to the annotation task we rely on various NLP tools and language resources. One important component is a Named Entity Recognition tool which detects entities in the documents that fall into a predefined set of semantic categories, like persons, locations, organizations, job titles etc. As a side-effect it takes care of some aspects of the sentences that a general parser might have trouble with, such as multi-word units and unknown words.

We have experimented with a range of Named Entity detection tools, e.g. ANNIE (included in the Gate[2] system) and LingPipe[3]. Both systems attempt to recognize the standard named entities like Person, Organization and Location. For the ANNIE system the number of detected types is slightly larger. Both systems also provide coreference resolution. Currently we are working with the CAFETIERE system (Black et al., 2003; Vasilakopoulos et al., 2004).

Another component that we rely upon is a dependency parser which has the purpose to detect the syntactic relations between the words. Dependency parsers output a set of named binary relations between words. Those relations can be regarded as a preliminary form of the annotation. Although they deal mostly with syntax (determining the relations of type subject, object, attribute etc), they can be regarded as underspecified semantic relations which have to be modified to specify their meaning in order to make them useful in the context of annotation.

Once syntactic dependency relations become available, it is necessary to normalize their structure. Normalization

[1] http://www.w3.org/2001/sw/

[2] http://gate.ac.uk/

[3] http://www.alias-i.com/lingpipe/

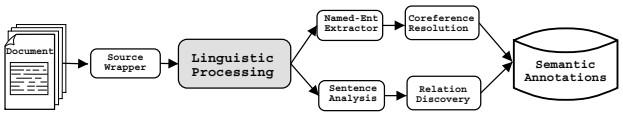


Figure 1: System architecture

applies certain transformations to the set of relations in order to unify their structure (e.g. in case different syntactic structures are semantically equivalent like in nominalization, copula use vs modifier use, active vs passive constructions, etc.).

Using the information from the Named Entity Recognition tool and the syntactic dependencies we obtain most of the information that we need. Combining both kinds of knowledge via a set of axioms lets us then calculate the semantic relations between the entities. Although currently we are experimenting with hand-crafted axioms, we plan to use a semantic resource as FrameNet (Baker et al., 2003), which would let us derive e.g. that "writing a book" is equal to "being an author of a book", thus further unifying the set of relations. The architecture of the system is displayed in figure 1.

The outcome of the automated annotation process might require (partial) human revision. This is supported by visualization tools which provide the means for the user to accept or reject a given annotation. We can visualize all the stages starting from the dependency structures up to the finished annotation. Figure 2. shows a visualization interface that allows the user to modify or correct the annotations via a web form.

2.1. Example

To illustrate more in detail the process of deriving the annotations, let us consider the following simplified example: "Novell, Inc. has acquired privately held Ximian of Boston, Mass., the leading provider of desktop and server solutions."

In the first step the Named Entity Recognition tool is capable of detecting the following entities, with their semantic types: "Novell Inc." (Organization), "Ximian" (Organization), "Boston, Mass." (Location). In the second step the syntactic dependency analysis detects the main verb "acquire" and its dependents, subject "Novell Inc." and object "Ximian", plus also a modifier-relation between "Ximian" and "Boston, Mass.".

Finally, in a third step, we can make use of an axiom that states that a modifier-relation between an Organization X and Location Y might be indicating that organization X is located at Y. In the case of the sample sentences, we aim at discovering the relations that can be paraphrased as: "Novell owns Ximian" and "Ximian is located in Boston".

3. Evaluation of dependency parsers

The analysis of a sentence in terms of a dependency grammar is a set of named binary relations between words. Although the relations mostly describe syntax (e.g. subject, object and attribute roles of the words), they can be regarded as underspecified semantic relations which have to be further modified to specify their meaning in the context of semantic annotation. Certain transformations are therefore needed to convert e.g. subjects and objects into semantic categories such as actor and patient. Obtaining these relations would be much more difficult in case of constituency-based structures.

There is no one dependency formalism. While different parsers treat the few main grammatical relations in a similar way, the remaining relations (e.g. adverbials) are handled rather differently. The number of different relation types varies a lot. Also some parsers attempt to describe deeper syntactic/semantic relations (e.g. control structures, passive subjects).

Partly due to the availability of large-scale language resources the reliability and coverage of dependency parsers have considerably grown in recent years. We evaluated 4 different parsers.

3.1. Connexor Machinese Syntax

Machinese Syntax[4] is a commercial rule-based parser based on the Functional Dependency Grammar formalism (Järvinen and Tapanainen, 1997). Its output is clearly surface-syntactic, for a more semantic output a different product (Machinese Semantics) is available[5]. Machinese Syntax uses a system of ∼36 dependency types. Different from the other parsers is the semantic distinction of adverbials ("time", "duration", "location", "manner").

Machinese Syntax is mature as a software product and comes with an API.

3.2. Link Parser

Link Parser (Sleator and Temperley, 1993) is a rule-based system implementing a Link Grammar formalism. Although it outputs the sentences as a set of binary relations, it violates some basic dependency grammar principles: link grammar links are not directed, the main verb of the sentence is not the sentence root, etc. These reasons, along with the fact that Link Parser uses a large and foggy system of link types (all together ∼110 types) requires a complex transformation mechanism to be implemented in order to convert the links into usable relations.

Link Parser is a mature software product, highly configurable and comes with an API, but the parser has not been actively developed in the last few years.

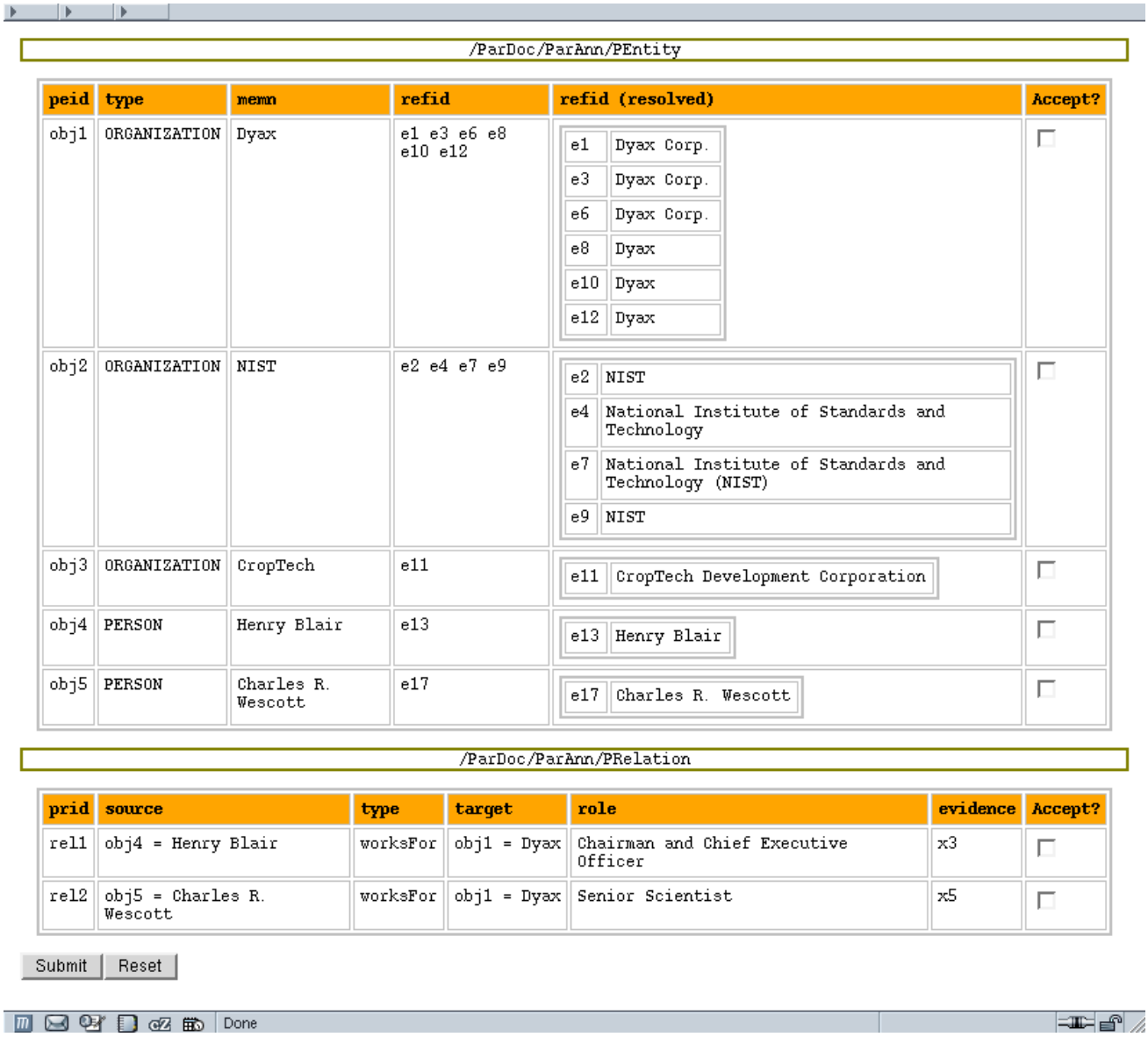
3.3. Pro3Gres

Pro3Gres (Schneider, 2003) is a rule-based parser which uses statistical information to rank and disambiguate the output. Probabilities are available for most of the main dependency types (in total, 24 different dependency types are used). It uses an external POS-tagger and a chunker, calculating the dependencies only between the heads of the chunks.

In addition to the traditional dependency output, Pro3Gres shows a Prolog-style predicate-argument structure for the input sentence. Often, it is more informative,

[4] http://www.connexor.com/m_syntax.html

[5] We only experimented with the web demos of Connexor's products. For the Machinese Semantics no such demo existed at the time of writing this paper



/ParDoc/ParAnn/PEntity

peid	type	memn	refid	refid (resolved)	Accept?
obj1	ORGANIZATION	Dyax	e1 e3 e6 e8 e10 e12	e1 Dyax Corp. e3 Dyax Corp. e6 Dyax Corp. e8 Dyax e10 Dyax e12 Dyax	☐
obj2	ORGANIZATION	NIST	e2 e4 e7 e9	e2 NIST e4 National Institute of Standards and Technology e7 National Institute of Standards and Technology (NIST) e9 NIST	☐
obj3	ORGANIZATION	CropTech	e11	e11 CropTech Development Corporation	☐
obj4	PERSON	Henry Blair	e13	e13 Henry Blair	☐
obj5	PERSON	Charles R. Wescott	e17	e17 Charles R. Wescott	☐

/ParDoc/ParAnn/PRelation

prid	source	type	target	role	evidence	Accept?
rel1	obj4 = Henry Blair	worksFor	obj1 = Dyax	Chairman and Chief Executive Officer	x3	☐
rel2	obj5 = Charles R. Wescott	worksFor	obj1 = Dyax	Senior Scientist	x5	☐

Submit Reset

Done

Figure 2: Visualization of entities and their relations. It can also be used as an interface that allows simple human revision by accept/reject

since it doesn't follow the dependency grammar restrictions (e.g. word can modify only 1 word). This output explicitly displays information about e.g. passive subjects, control structures.

3.4. MINIPAR

MINIPAR (Lin, 2003) uses internally constituency-structures (where each subtree has a head), but outputs dependency structures. It is similar to Pro3Gres since it is rule-based and uses statistical information extensively to disambiguate between the readings. Also, it attempts to output a more semantic reading of the sentences, showing the traces involved in the relations. It also tries to detect a small number of named entities (person, title). The number of different relation types used is ∼35.

3.5. Evaluation

In the evaluation, we didn't focus on the coverage and accuracy of the parsers. This has been tested by (Mollá and Hutchinson, 2003; Lin, 2003; Schneider, 2003) on the bases of the subset of 500 sentences from the SUSANNE corpus, which has been manually analyzed to provide it with dependency information (Carroll et al., 2003). According to this existing work, all the parsers show equal results, only Link Parser's results tend to be lower.

Rather, we focused on the usefulness of the parsers as a component of an automatic annotation system. Important, therefore, is the functional nature of the output (i.e. how difficult would it be to implement a transformations layer), but also the development status, robustness and configurability of the parsers. Our intention is to apply dependency parsing to already preprocessed input for which named entity information exists. Therefore it's important that we can configure the parser in a way that it could make use of this information and would not e.g. attempt to analyze the internal structure of the entities.

We can conclude that no clear winner exists. While Machinese Syntax and Link Parser are mature as soft-

ware projects, MINIPAR and Pro3Gres generate as output a structure which can be better used as a component of a semantic annotation system.

4. Combining Named Entities and syntactic dependencies

The arguments of dependency relations as output by a dependency parser are words (plus maybe their POS tags). This information is not enough to build semantic annotations. Named Entity Recognition plays a crucial role here as the arguments of a dependency relation can be rendered semantically more informative using entity types. Therefore, combining both kinds of knowledge lets us calculate the semantic relations between the entities.

Abundant in e.g. news documents are cases where the dependency relation (and the head of the dependencies) is not informative enough, e.g. the sentence "Smith, CEO of Lang, Inc. has announced a merger with Mind, Boston, Mass" contains several dependencies of type "mod", where the head is a comma or a preposition ($Smith \leftarrow^{mod} of \rightarrow^{mod} Lang$). By a set of axioms that specify the relations, we can still recover the true (semantic) nature of the dependency.

$$Person(x) \wedge Organization(y) \wedge mod(x,y) \Rightarrow worksFor(x,y)$$

Figure 3: An example of an axiom.

An example axiom in figure 3 states that if x is of type $Person$ and y of type $Organization$ and x and y are syntactically related through modification then x works for y i.e. through this transformation the relation "mod" is replaced by a more specific relation "worksFor".

Different dependency structures can stand for the same relation (e.g. $AuthorOf(X,Y) = Wrote(X,Y)$, $WorkFor(X,Y) = Employ(Y,X)$). The creation of the axioms that would handle such equalities is a potential bottleneck of the proposed approach, which might be solved by exploiting resources such as FrameNet. These equalities could also be calculated automatically as described e.g. in (Lin and Pantel, 2001).

5. Conclusions and future work

In this paper we have presented some aspects of our work aimed at combining syntactic knowledge from dependency-based parsing and semantic knowledge about entities for the purpose of semi-automatic creation of semantic annotations. This is a very ambitious goal, and many hurdles stand in our way. In particular the set of axioms which are needed for the 'normalization' of the generated logical forms, and the ontologies to be used, are still a major issue. We are currently working on those problems.

Acknowledgments

The authors gratefully acknowledge the support of the Parmenides project (EU project, contract No. IST-2001-39023) and the Swiss Federal Office for Education and Science (BBW/OFES). The first author was supported by an ESKAS grant (Eidgenössische Stipendienkommission für ausländische Studierende).

6. References

Baker, Collin F., Charles J. Fillmore, and Beau Cronin, 2003. The Structure of the Framenet Database. *International Journal of Lexicography*, 16.3:281–296.

Black, William J., John McNaught, Argyris Vasilakopoulos, Kalliopi Zervanou, Babis Theodoulidis, and Fabio Rinaldi, 2003. CAFETIERE: Conceptual Annotations for Facts, Events, Terms, Individual Entities, and RElations. Technical Report TR-U4.3.1, Department of Computation, UMIST, Manchester.

Buitelaar, Paul and Thierry Declerck, 2003. Linguistic annotation for the semantic web. In (Handschuh and Staab, 2003).

Carroll, John, Guido Minnen, and Ted Briscoe, 2003. Parser evaluation: using a grammatical relation annotation scheme. In Anne Abeillé (ed.), *Building and using Parsed Corpora*. Dordrecht: Kluwer.

Cimiano, Philipp and Siegfried Handschuh, 2003. Ontology-based linguistic annotation. In *The ACL-2003 workshop on Linguistic Annotation*. Sapporo, Japan.

Connolly, Dan, Frank van Harmelen, Ian Horrocks, Deborah L. McGuinness, Peter F. Patel-Schneider, and Lynn Andrea Stein, 2001. DAML+OIL (March 2001) Reference Description. http://www.w3.org/TR/daml+oil-reference.

Handschuh, Siegfried and Steffen Staab (eds.), 2003. *Annotation for the Semantic Web*. IOS Press.

Järvinen, Timo and Pasi Tapanainen, 1997. A Dependency Parser for English. Technical report, Department of General Linguistics, University of Helsinki.

Lin, Dekang, 2003. Dependency-based evaluation of MINIPAR. In Anne Abeillé (ed.), *Building and using Parsed Corpora*. Dordrecht: Kluwer.

Lin, Dekang and Patrick Pantel, 2001. Discovery of Inference Rules for Question Answering. *Natural Language Engineering*, 7(4):343–360.

McGuinness, Deborah L. and Frank van Harmelen (eds.), 2004. OWL Web Ontology Language, Overview, W3C Recommendation 10 February 2004. http://www.w3.org/TR/2004/REC-owl-features-20040210/.

Mollá, Diego and Ben Hutchinson, 2003. Intrinsic versus Extrinsic Evaluations of Parsing Systems. In *Proceedings European Association for Computational Linguistics (EACL), workshop on Evaluation Initiatives in Natural Language Processing*. Budapest.

Rinaldi, Fabio, James Dowdall, Michael Hess, Jeremy Ellman, Gian Piero Zarri, Andreas Persidis, Luc Bernard, and Haralampos Karanikas, 2003a. Multilayer Annotations in PARMENIDES. In *The K-CAP2003 workshop on "Knowledge Markup and Semantic Annotation"*.

Rinaldi, Fabio, Kaarel Kaljurand, James Dowdall, and Michael Hess, 2003b. Breaking the deadlock. In *ODBASE 2003 (International Conference on Ontologies, Databases and Applications of SEmantics) Catania, Italy.*, volume 2889 of *Lecture Notes in CS*. Springer Verlag.

Schneider, Gerold, 2003. Extracting and Using Trace-Free Functional Dependencies from the Penn Treebank to Reduce Parsing Complexity. In *Proceedings of TLT 2003*. Växjö, Sweden.

Sleator, Daniel and Davy Temperley, 1993. Parsing English with a Link Grammar. In *Third International Workshop on Parsing Technologies*. pages 277–292.

Vasilakopoulos, Argyris, Michele Bersani, and William J. Black, 2004. A Suite of Tools for Marking Up Textual Data for Temporal Text Mining Scenarios. In *Proceedings of the 4th LREC Conference, Lisbon*. (accepted for publication).